〈論　文〉

地場企業におけるソフトウエア工学活用状況

荒　川　淳　三

1．はじめに

ソフトウェア工学はソフトウェアを工業的に生産するための技術である。1968年に始まる"GO-TO-LESS"プログラムの論争以降，品質の優れたプログラムを如何にして安価に，短い工期で生産するかについて，さまざまな提案がなされてきた。ウォーターフォール型ライフサイクル，構造化分析，データフローに基づくシステム設計，データタイプに基づくシステム設計，プロトタイピング，レビュー，ウォークスル，チーフプログラマチーム，各種開発支援環境などが代表的な例である。

これらの技術はソフトウェア開発に威力を発揮し，例えば1970年ころ大きな問題となったソフトウェア危機（開発対象が大きくなるにつれソフトウェア開発は加速的に困難になるので開発可能なシステム規模は自ずと制限されるという危機感）もこれによって乗り越えることができた。

しかし今日のソフトウェア工学は次のような問題を抱えている。

(1) マイクロエレクトロニクスの進歩がもたらしたコンピュータハードウェアおよび通信技術の驚異的な進歩がソフトウェア開発の需要を巨大化したため，ソフトウェア業界の開発能力がこれに対応できない。

(2) ソフトウェア工学の成果が，ソフトウェアの生産現場で十分に生かされていない（という主張がしばしばなされる）。

本論文はソフトウェア生産の現場におけるソフトウェア工学の適用状況についての基礎的調査である。調査の焦点は次の諸点である。

(1) ソフトウェア工学の成果としての個々の技法や方法論は，ソフトウェア生産の現場でどの程度利用されているか。

(2) ソフトウェア工学自体に，その充分な適用を阻害している要因はないか。

(3) ソフトウェア企業側に問題はないか。

(4) 日本の先進的企業と札幌近辺の地場産業とを比較するとき，ソフトウェア生産技術全般あるいはソフトウェア工学の活用状況にどんな差があるか。後者は前者に何を学ぶことができるか。

調査に当たっては東京，大阪地区の先進的ソフトウェア企業2社と北海道ソフトウェア企業4社を選び，これら企業に対してアンケートおよび面談によってのソフトウェア技術に関する調査を行い，結果を分析した。

2．調査の方法

2.1　ソフトウェア企業の選択

本調査は私の最初の調査であり，基礎調査として性格づけた。調査対象企業は計画段階から数社に限定したが，最終的に選択した企業の一覧を表-1に示す。

表のうちA社にはアンケート調査に対する協力を得られなかったが，面談には応じて頂いた。D社はライフサイクルの個々のフェーズにおける適用技術の調査には回答してくれなかったが，これは同社が取り引きを行っている大口の顧客に配慮しての結果である。D社の受注する業務のかなりの部分は顧客の技

表－1 調査企業の一覧

企業名	企業立地	企業の性格	選択の理由
A	東京	ユーザ系	日本を代表する総合的ソフトウェア企業
B	大阪	ユーザ系	最近分社独立、積極的な技術方針を持つ
C	道内	ユーザ系	親企業への依存率が比較的高い
D	道内	ユーザ系	親企業への依存度が比較的低い
E	道内	ユーザ系	製造業から分社独立、多角的経営を展開
F	道内	独立系	独立系の中堅企業

術標準に従って行われるので，この技術を公開することは顧客の技術を公開することにつながる恐れがあるからである。

今回の調査ではコンピュータメーカー系の企業を選ばなかったが，その理由はメーカー系の企業では本調査が主眼を置くビジネスシステムの開発は行っていないと考えたからに外ならない。

2.2 アンケート調査の項目

表-2はアンケート用紙の内容とそれについての若干の注釈とである。アンケート用紙の記入用余白は，紙面の効率化のためにここでは削除してある。

アンケートは大きく四つの部分にわかれるが，最初の部分は企業の特性を知るための一般的な質問である。ここではシステムアナリストをどう定義するか，プロジェクトチームのメンバー数はプロジェクトのどのタイミングでとらえるかなど，ややあいまいな点が含まれているが，これについてはアンケート記入まえ，記入後の2回の面談で可能な範囲で確認した。

アンケートの次の部分は企業が日頃遭遇している問題点とその対策についてである。ソフトウェア開発技術の細部に直接入ってしまうと「木を見て森を見ない」ことが心配されるため，この部分を設けた。

ここで取り上げた15の項目のうち，最初の販売促進活動から第10項のプロジェクト管理までは，次の「ソフトウェア工学的技術活用状況」と項目，順番ともに一致するが，ここでは以上のほかにプロジェクト品質保証，プロジェクト納期保証，技術蓄積と再利用，社員教育，新製品開発・技術開発の総合技術的性格を持つ5項目を付加している。また15項目について，問題の重要度による優先順位づけをお願いしている。

ソフトウェア工学的技術活用状況では，プロジェクトライフサイクルの10のフェーズについて，列挙された技術，あるいは方法論の中から実際に活用しているものを示し，その効果を確認するという方法を採った。

最後に，特に付記すべき事項の記述欄を設けた。

3．調査結果

アンケートに対する6社からの回答は表.3にまとめた通りである。表中の項目ごとに付してある注記は，各社との面談によって確認した事項が主である。

4．考　察

以下に調査結果に基づき，考察を試みる。考察はアンケートIIでとりあげた15の項目について行う。

4.1 販売促進活動

A社は他に比べると早い時期にユーザ企業から分社し，総合的なソフトウェア企業に成長しているため，強力な営業体制を備えている。

B社は分社して日が浅く，親企業の仕事の比重がかなり高い。このためA社と比べると営業要員の比率が低いが，営業活動の重要性を充分に認識しており，本項を最重要課題としている。親企業が製造業であるため，業務診断に熟達したIE技術者を抱えており，このメンバーの活動が販売促進に直結する強みをもっている。

C社もユーザ企業から分社したソフトウェア企業であるが，親企業への依存度が高く，また親企業以外には特定のコンピュータメーカーを顧客としているため，販売促進活動には特に問題を感じておらず，営業要員は極めて少数である。

D社は親企業への依存度は低く，新しいマーケットの開拓に積極的であるが，特定のコンピュータメーカー，特定のユーザ企業が主な顧客となっており，積極的な販売促進活動を必要としていない。営業体制は強力ではない。

E社はB社によく似ており，IE技術が営業促進の武器になっている。営業体制は道内他社に比べるとはるかに強力である。

F社は特定のソフトウェアハウスとの取り引きが大半であり，販売促進活動は特に行っていない。営業体制は弱体である。

以上を集約すると次の主張が成り立つと考える。

(1) 完全に独立したソフトウェア企業は強力な営業促進体制を必要とする。A社の2000人中200人の営業部隊というのは一つの指標である。

(2) 製造業の社内で広く行われている業務診断は，一般のマーケットを開拓するための強力な武器となる。製造業から分社したソフトウェア企業は，この点で有利である。

(3) 道内のソフトウェア需要は貧弱であるが，関東，関西等に手を伸ばせば充分な注文が得られる。この際特定のソフトウェア企業，コンピュータメーカー等から連続した注文を得られれば，営業活動上も技術的にも（馴れという意味で）有利であるが，この方式は営業力の弱体化，さらには企業の系列化につながる恐れが多分にある。道内ソフトウェア企業の営業体制は平均的にかなり弱体であり，強化される方向にもない。

4.2 見積り

見積り作業の現状は次のように要約できる。

(1) 注文対象を詳細に分析し，要素ごとの推定値を積み上げる方式は，次の理由で成り立たない。

- システム分析のフェーズまで含む膨大な作業を行わないと，すなわち実務に大きく立ち入らないと，業務要素が自信の持てる信頼度で確定しない。
- 要素を充分に確定しても，得られる見積り精度が不十分である。

(2) 過去の実績からマクロ的に推定する方式で得られる精度は充分ではないが，積み上げ方式よりむしろ高い。

(3) 業務内容，開発環境，使用言語を含む開発技術等で未経験の要素が加わると見積り精度が低下するが，発注者側の技術標準に従って開発を行う契約（大規模ユーザ，コンピュータメーカーなどとの契約では一般的である）では，この種要素に振り回されることが多い。また同一プロジェクトを多数のソフトウェア企業が参加して行う場合，インターフェースの難しさが見積り誤差の原因となる。

(4) 上記諸要素を取り込んで見積りを行えば精度は向上する筈で，このような見積りモデルを検討している企業もある。

(5) ファンクションポイント法を含めて，決め手となる技術は存在しない。

(6) 現状で見積りの狂いによる被害を逃れるには，実績によって精算する方式，業務を複数段階に分割して先ず最初の段階のみの見積りを行い，前段で得られた精度の高いデータ

に基づいて次段の見積りを行う方式などが考えられ，検討あるいは実行されている。

結論的に，見積りの問題をきちんと解決する具体的方策は，次のように考えられる。

(1) 自社で獲得した注文を自社の技術で開発する体制を確立し，自社の専門領域を限定することにより，実績を効果的に蓄積する。

(2) ソフトウェア開発のコスト，工期を目覚ましく改善し，適正な技術料等を含めた余裕のある見積りを可能にする。

4.3 要求仕様の確定

要求仕様の確定に関する問題点は次のように集約できる。

(1) 商取り引きの一環としてのルールが確立されていない。顧客側の体制と仕様確定までのプロセス，仕様変更時責任の契約上の明確化など。

(2) ユーザ側の力不足

(3) SE の業務知識不足

(4) まとめられた要求仕様が必要にして充分であることに確信が持てない

(5) 要求仕様が開発着手後に変更される

(6) システムインテグレーター的企業が間に入るときの自社としての仕様，スケジュール等の確定の手順，体制が難しい

(7) 要求仕様の再利用がうまく行われていない

上記に対して採られている対策，技法は次の通りである。

(1) あいまいさを残さない契約方法の検討

(2) ウォークスルーは広く有効に使われているが，どこまで「構造化」されているかは不明である

(3) B社は構造化分析を標準的技術として位置づけ，その効果を確認しており，E社も構造化分析技術を試行し，効果確認の途中である。いずれの場合にも構造化支援ツールが導入されている。

(4) B社はプロトタイピング，デシジョンテーブルも効果的に活用しており，エンティティリレーション分析についても効果確認の過程にある。

(5) SE の業務知識強化の必要性は各社が認識しているが決め手を欠いている。

(6) 要求仕様の蓄積再利用は，B社，E社において検討されている。

以上から，要求仕様の確定は，基本的に次の方向に動くものと考えられる。

(1) 構造化分析技法の普及定着

構造化分析技法，ツールで作成された諸ドキュメント（構造化仕様書）は可視性にすぐれ，蓄積再利用にも好適であるので，この活用が普及し，普及とともにその再利用技術も蓄積されるであろう。

(2) SE 業務知識の強化は，要求仕様確定技術がソフトウェア企業にとっての本来的最重要技術であると認識することによって進むであろう。

(3) SE 業務知識の強化，要求仕様技術の蓄積再利用いずれもが，対象業務領域を限定することで一挙に容易になる。この検討が真剣になされるべきである。

(4) プロトタイピングは殆ど活用されていない（B社のみ）が，要求仕様確定の重要性を考えるとき，いささか意外である。後述の第4世代言語同様，中小型の世界で先に一般化することになるのかも知れない。

(5) 自社の営業活動が弱い企業は，この分野で遅れをとることが危惧される。

4.4 システム設計

システム設計に関しての各社の問題意識は次のように要約できる。

(1) 設計作業が個々人の経験に基づいて行われている。共通的な方法論に裏打ちされていないと設計思想の統一がなされず，設計の再利用にも不利である。

(2) 顧客の設計基準を適用することを要求され，自社の基準で作業できない。

(3)　設計の評価が困難である。
(4)　設計の再利用が充分になされていない。
(5)　分散化等の新技術への追従が充分にできていない。
(6)　ドキュメントの作成，修正の負荷が膨大である。

これらの問題にたいして，次のような対策が講じられている。
(1)　プログラム構造のパターン化はほとんどの企業で活用され，生産性，品質の向上に寄与している。
(2)　設計査閲，ウォークスルーはB社，E社等で活用され，システム品質の向上に効果をあげており，教育上の効果も大きい。
(3)　設計における特定の方法論はB社が検討しており，E社では経験と融合させあまり意識することなく実作業に活かしている。
(4)　B社，E社は自動化支援ツールの活用も一部行っている。

設計は本来経験の再利用が効果的な作業であり，特定の方法論を個々の作業に適用するには馴染まない。構造化設計技法等がほとんど利用されていない現実はこれに起因すると思われる。

しかし後の基準になる設計は標準的な方法論に基づいて行われなければならない。

方法論によって設計思想を統一し，この思想で作られた標準設計を電子ツール上に蓄積し，これらを効果的に再利用しながら設計作業を進めるというのが，現在考えられている目標である。

一連の技術において，B社，E社は，道内のC社，D社，F社よりも，技術的に明らかに先行しているように見える。自社の営業体制がそれなりに強化され，自社の技術で業務を展開する機会に恵まれていることが，その理由と思われる。

4.5　プログラム設計

設計書の書き方が不統一である，設計書の評価が難しい，設計書をどの程度の詳細さで書くべきか，設計書はそのままでは保守用ドキュメントとして使えないなどの問題があるが，上流工程に比すれば問題は小さい。

欧米では「設計書をどの程度の詳細さで書くべきか」という問題は生じないが（欧米では基本的には全論理を設計記述言語で書く），プログラマを初級SEに，さらには上級SEに育てていく日本では，これが重要な問題になる。アメリカ産のソフトウェア工学の直輸入ではなく，日本の実情に合わせて見直されたソフトウェア工学が必要な理由の一端を垣間見ることができる。

設計図法は広く使われるようになっており，この結果構造化プログラミングの定着は顕著である。プログラム構造のパターン化は殆どの企業で採用され，プログラム品質，生産性の両面で効果をあげている。

B社，E社，F社では設計図の作成，設計図とコード間の双方向自動変換を支援するツールが導入され，活用されている。

ドキュメント支援ツールの本格的導入は今後の課題である。

4.6　コーディング・単体テスト

個人の能力差が大きい，標準化が進まない，単体テストが充分に行われていない，プログラム設計者とコーダー間の意志疎通が充分でないなどの問題があるが，プログラム設計より更に問題は小さい。

コーダーに対する指導体制の強化，コードのパターン化，プログラム設計図からの自動変換（この場合にはプログラム設計書が完全に記述されなければならない），構造化プログラミングの徹底などが，上記問題点の対策として実施され，効果をあげている。

第4世代言語はほとんど利用されていない。しかし中小型コンピュータ関係では例えばD社の場合，状況が一変する。AS400を4年リースで導入するような例が多く，極めて

短期の納期が要請されるため，既存パッケージの活用が必須であり，言語はRPGとなる。

今後予想される中小型への大幅な移行を考えると，ソフトウェアの革命は中小型の世界から進むことも考えられる。

4.7 システムテスト

長時間を要する，負荷が大きい，テスト環境が作りにくくテストの質が充分でない，要求仕様書を軸とした客観的なテスト方式が確立されていない，先行工程のスケジュール的しわ寄せをうけるなどの問題が提起されているが，重要度の評価は低い。

一般のプロジェクトにおいてはシステムテストの作業量が全体の半分近くを占め，構造化技術の徹底によりこの部分を劇的に減少させることが可能であるとの報告(1)もなされていることを考えると，若干奇異な感じがする。

テスト環境の早期整備，第3者テストの実施，テストケースに対するウォークスルーの実施などが対策として行われている。

プロジェクトの品質，コスト等が詳細に分析されるようになると，現状よりはるかに大きい関心がシステムテストに集まるようになろう。

4.8 インストール・引渡テスト

インストール，引渡テストに関する問題点としては，インストール時間を極力短縮することが要求される，この段階でユーザの仕様変更が多発する，移行漏れが生ずるなどが挙げられているが，大きな問題として認識はされていない。

対策としてはファイルコンバージョン結果の自動判定が挙げられているが，これは各社が行っていることと思われる。

インストールがあまり大きな問題になっていない理由としては，大規模アプリケーションシステムを納入する機会が少ないと思われること，日本では契約を厳格に貫く習慣が弱いことなど考えられる。

4.9 システムメンテナンス

ドキュメント不備のプログラム，非構造的なスパゲッティプログラム，アセンブラプログラムの混在，結果的に生じるメンテナンス要員のはりつけと士気の低下，エンドユーザとの直接取り引きでけじめのつかないメンテナンス作業，メンテナンスの影響範囲の特定化困難など，ここでは昔からの問題が解決されず，拡大する一方である。

対策としては決め手を欠いており，ドキュメント管理の強化，データのクロスレファレンス機能の支援などが行われている程度である。アセンブラ，あるいは非構造的プログラムを構造化された第3世代言語に自動変換するエンジンは強く求められているが，具体的な動きにはなっていない。

システムメンテナンスがこのように陽の当たらない世界になっている理由としては，メンテナンス費用がソフトウェアのライフサイクルコストの中で正確に把握されていない（場合によっては開発予算の不足をメンテナンス作業で補うこともある），システム機能の追加変更が，本質的にはシステム開発作業と同じであることが理解されていないなどを挙げることができる。

今日のソフトウェアの世界は急速に動いており，ユーザ企業の情報システム部門は，開発部門から問題解決促進部門へ，情報処理主体から情報資源管理主体へと体質を変えようとしているが，システムメンテナンスが旧態依然たる形で残ることは，この動きに対して非常な障害になる。新しい体制では「ソフトウェア企業に開発技術がある」ことが必要になるが(技術は専門化によって進歩する)，メンテナンスがユーザ企業にのこる限り技術の完全な移動が困難だからである。

近い将来，システムメンテナンスに鋭いメスが入れられ，システム機能の追加変更作業

が，システム開発作業同様，外注化されることが必要である。

4.10　プロジェクト管理

プロジェクトメンバーにプロジェクト管理についての共通認識が乏しくプロジェクト管理の実行力も不足している，プロジェクト管理のできる人材が不足している，多数のソフトウェア会社が参加するプロジェクトでの自社責任が不明瞭である，若年層に大規模プロジェクトを経験する機会が少ないなどの問題があり，各社とも問題を重視している。

対策としてはプロジェクト管理教育が種々工夫されているが，プロジェクト管理をソフトウェア技術者の基礎的素養と考え，新入社員教育からプロジェクト管理の教育を行うというB社の考え方は注目に値する。E社が大規模プロジェクトの展開過程を蓄積保存し，これを教材にして若い技術者に疑似体験を与えようとしているのも注目される。

F社はプロジェクトの進捗管理に自動支援ツールを導入している。

プロジェクト管理はプロジェクト成否の死命を制する重要な技術であり，ソフトウェアプロジェクトの流通商品化が進むにつれて，益々注目を集めることになろう。ここにおいても要は技術の蓄積と再利用である。

4.11　プロジェクト品質保証

プロジェクトの品質保証については，各社ともその重要性を強く認識している（C社は1位）が，具体的な活動はあまり進んでいない。

重要と考えるが手がまわっていない，品質保証のための統一的な制度が設定されていない，上流工程での品質確保とプログラム品質の検査方法確立が課題であるなどの問題認識がなされ，ウォークスルー，フェーズドアプローチの徹底，品質管理体制の確立等が進められている。

品質保証は要求仕様の確定から引渡テストまでの全ての工程の集積であり，独立した存在ではない。品質保証が確立されたときはソフトウェアの生産技術が確立されたときであると言える。

4.12　プロジェクト納期保証

納期については見積り不備，環境変化，プロジェクト管理不徹底，品質評価を含めての進捗管理の未確立等の問題が認識され，フェーズドアプローチ，ウォークスルー等の徹底が対策として行われている。

しかし品質保証同様，この問題にまで充分に手がまわっていないと言える。今日のソフトウェアプロジェクトに必要とされる工期は，この技術革新の時代においては非常識なほどに長いものである。

プログラムを新たに作る限り正確な見積りは出来ず，品質保証も納期保証も不可能であるというのが今日の技術水準であり，これらを根本的に解決するためには，既存ソフトウェアを徹底的に再利用するソフトウェア生産技術の確立が不可欠である。

4.13　技術蓄積と再利用

再利用がソフトウェア生産技術の基本であり極めて重要であること，蓄積再利用されるべき対象がシステム機能，システム構造，モジュール構造，モジュール部品，データ，プロジェクト管理などの全てに及ぶことは各社とも理解しているが，現実は全く不十分であることも各社の等しく認めるところである。

B社は蓄積対象の個々が統一的方法論に貫かれていることの必要性を認識し，これに向かって行動を開始している。専門組織の設置，対象領域の限定，支援ツールの導入などの必要性が理解されている。

さまざまなソフトウェア部品を蓄積することは容易であるが，再利用可能な部品の蓄積となるとことは簡単ではない。過去の例でも

部品（広い意味での）の蓄積再利用に成功した企業では，これを可能にするための技術基盤の整理（例えば情報隠蔽技術の徹底）と蓄積体制の試行錯誤の繰り返し等の苦難をくぐり抜けている。

技術蓄積と再利用は，技術体制確立の正念場である。

4.14 社員教育

社員教育の重要性は各社が充分に承知しており，それに高い優先度を与えている。実践教育に偏った現状を改め体系的な教育を行うこと，自己啓発へのインセンティブの強化などの必要性が認識されているが，技術の蓄積再利用のような高度な技術体制の確立を目指すときには一貫した技術体系（discipline）基盤が必要になり，これはB社の指摘するところである。

方法論の体系を基盤として技術の蓄積再利用が行われるのは工学の基本であるが，ソフトウェア生産に関してはこのような基盤が脆弱である。

ソフトウェア生産の発展過程においてはさまざまな問題が現れ，その対策が方法論として整備され体系化されてきた。代表的な例をあげれば，優れたプログラマと未熟なプログラマの差は"GO-TO-LESS"の問題として提起され，これに対しては構造化定理によって裏打ちされた構造化プログラミング技術が確立された。またBrooksの提起したソフトウェア生産における分業の困難さについてはフェーズドアプローチとフェーズ間でのレビュー，ウォークスルー技術，モジュール化技術等が考案された。

ソフトウェア生産を電気設備，機械設備等の生産と同様な工学のレベルに引き上げるためには，学校教育と企業内教育が有機的に体系化されねばならない。大学教育でソフトウェア工学の体系を学び，企業内教育で実践的に鍛えられたSEがソフトウェア生産の主流となるとき，ソフトウェア生産の技術は一段と高度化すると思われる。

SEの育成において大学と産業界が協力しあい，調和のとれた教育が展開されるためには，おそらく大学教員には広範囲の学習が必要となる。ソフトウェアを教える大学教員のほとんどはソフトウェア生産作業の実態やそこで生じている問題を理解していないし，ソフトウェア工学の基本も身につけていない。例えばフォートラン言語を用いたプログラミング教育の多くは命令語を組み合わせて求められた機能を実現することに終始し，暗黙の型宣言さえ用いられている。データについての説明や適切なコメントの記入などが重要視されない実態は，大学で用いられている代表的なフォートランのテキストを見れば一目瞭然である。文科系のプログラミング教育であればシステム設計，プログラム設計などは全く教えられない。

ソフトウェア業界から「大学でのコンピュータ教育など役に立たない」という発言が現実になされるが，これは大学におけるコンピュータ教育の問題と，ソフトウェア業界の管理者の問題を同時に表現しているものであろう。

4.15 新製品開発・技術開発

製品開発，技術開発はマーケッティング力を背景になされるのであり，マーケッティング力の強化が先ず必要である，プログラム間の変換エンジンが必要であるとの意見が出されているが，ここには殆ど手がまわっていないのが実態であろう。

5．結　論

5.1 道内ソフトウェア企業の技術的特性

(1) 注文の多くを北海道外の大手ソフトウェア企業あるいはコンピュータメーカーから安定的に得ているため，自力でマーケットを拡大するための営業力が弱体である。

(2)　営業力については，製造業から分社した企業に一般的な業務診断能力が，販売促進上の有力な武器となっている。業務診断技術の蓄積に乏しい企業は，自己の弱点としての自覚が必要と思われる。

(3)　上記注文の多くが従うべき技術標準を指定するため，自社の技術蓄積が充分に行えない。

(4)　企業が円滑に成長していくには営業力，技術力，製造力がバランスし，自己発展的に機能しなければならない。この意味において北海道内のソフトウェア企業の多くは，将来性に疑問がある。

(5)　以上の一つの帰結として，道内ソフトウェア企業の技術的問題についての認識は，道外の先進的ソフトウェア企業に比べると遅れている。

(6)　ソフトウェア工学の諸技術についての道内企業の現状は次のように集約できる。

・構造化ウォークスルー，レビュー
　広く利用されているが，どこまで厳格に手順化され活かされているかは不明である。
・構造化分析（技法とツール）
　道内においても先進的な企業はその活用に着手しているが，一般の企業はそこまで至っていない。
・プロトタイピング，エンティティリレーション分析
　先進的企業では一部利用されているが，道内では利用されていない。
・構造化設計技法，ジャクソン法，ワーニエ法
　利用されていない。
・システム構造のパターン化
　広く利用され，プログラム品質，生産性の向上に効果をあげている。
・設計図法
　広く利用されている。結果的に構造化プログラミングの徹底に繋がっている。
・プログラミングツール
　プログラム設計図とコードとの双方向自動変換ツールが普及しつつある。
・プログラム構造のパターン化
　広く利用され効果をあげている。
・モジュールの部品化
　利用されていない。
・パッケージのカスタマイズ
　利用されていない。ただし中小型コンピュータの世界では広く利用されている。
・第4世代言語
　利用されていない。ただし中小型コンピュータの世界では広く利用されている。
・トップダウンテスト
　利用されていない。
・データのクロスレファレンス
　かなり利用されている。
・プログラム構造化エンジン
　利用されていない（実用化されていない）。
・ソフトウェア資産管理
　モジュール単位で履歴の管理を行う厳格な資産管理は行われていない。
・プロジェクト管理用ツール
　一部進捗管理に適用され，効果をあげている。
・ソフトウェアの蓄積，再利用
　プロジェクト実績，プログラム構造，システム構造など，局部的に行われているが，一貫開発支援ツールを基盤とした体系的な蓄積再利用はこれからの課題である。

5.2　ソフトウェア企業の技術的活性化対策

(1)　好不況に左右されない安定成長を考えるとき「蓄積なき経営」は重大な問題であり，製造力のみでなく，技術力，営業力等の常日頃の蓄積を図らねばならない。

(2)　上記に関しては同業他社等との日頃の切磋琢磨が必要であり，中堅管理者を対象とした地域交流の活発化が必要である。

(3)　構造化分析，エンティティリレーション分析，プロトタイピング，種々の開発支援ツー

ル，構造化エンジンなどの重要技術については，地域としての教育普及体制が必要ではなかろうか。

きちんと学習する機会がないために新技術採用に乗り遅れているケースもあるように感じられる。

超多忙のSE，中堅管理者等を対象とする教育であれば，週日の夜間，あるいは休日等を利用して適正な頻度で行われなければならない。

(4) 限定された領域への特化

道内企業の抱える最大の技術問題は次の2つであろう。

(1) SEの業務知識不十分

② 技術の蓄積再利用が困難

これらのいずれもが，ソフトウェア企業が特定の領域（例えば大学事務支援システム）に専門化することで解決する。技術とは本来そういうものであり，ソフトウェア企業の多くが専門領域を持たないというのは，アプリケーションソフトウェア産業が未成熟である証左でもある。

5.3 ソフトウェア教育

(1) 今日のソフトウェア技術者の多くは，専門教育を受けた専門家ではない。電気技術者や機械技術者の育成過程と比べるとき，このことは明らかである。

(2) ソフトウェア生産を工学の領域にまで高め，ソフトウェアの品質，納期，価格を抜本的に改善するためには，大学時代からの一貫した工学的教育が行われなければならない。

(3) この第一歩として，大学のさまざまな課程でソフトウェアを教える教員の再学習が必要である。ソフトウェアの生産がどのような問題を抱えているのか，これを解決するための技術はどこまで来ているか等についての理解を得るためである。

(4) ソフトウェア教育についての産学協同体制が強化されねばならない。双方からの意見を吸収してのカリキュラムの確立が求められる。

(5) 「管理は計測から始まる」が，ソフトウェア教育を効果的に行うためには，習得結果の計測が必要である。加えてプロジェクト管理の円滑な展開のためにはSE能力の判定が極めて重要であることを主張する企業が多い。

「SE能力判定技術」の開発は，これだけでプロジェクトを設定する価値がある。

5.4 その他

(1) 日本的ソフトウェア工学の追求

「米国で広く使われているソフトウェア工学が日本では何故使われないのか」という問題提起がよくなされるが，この背景にはソフトウェアの生産を取り巻く日米間の社会体質の差があることは明らかである。

日本と異なり，米国はギルド的社会であり契約社会である。日本ではプログラマは能力を増すとともに初級SE，上級SE，システムアナリストと格付けを上げていくが，米国ではプログラマは（大学で資格を取るなどしなければ）いつまでもプログラマであり，アナリストは大学卒の新人でもアナリストである。日本でのプロジェクトは所期の目的を達成することを目標に展開されるが，米国でのプロジェクトは契約の履行が目標となる。この差がソフトウェア工学に与える影響は極めて大きいと考えられる。

米国で広く使われているソフトウェア工学が日本では何故使われないのかという議論はこの辺で打切り，ソフトウェアの生産に関して日米間で何が同じか，何が異なるかを充分に見極めた上で，日本の実情に合ったソフトウェア工学の展開がなされるべきである。

(2) ソフトウェアメンテナンスの在り方

ソフトウェア企業のみにかかわる問題ではないが，ソフトウェアメンテナンスの問題にメスを入れ，この在り方を変えなければならないと考える。機械設備等と異なり，ソフト

ウェアには劣化はない。システム機能の追加変更は，規模の大小にかかわらず，システム開発と本質は同じであり，技術体制の進歩とともに外注化されるべきものである。

ソフトウェアメンテナンスがユーザで行われるかぎり，ソフトウェア開発技術の専門企業への移転は不完全にしか行えず，技術進歩が阻害されるからである。

6．おわりに

本調査を終えての反省点の１つは，アンケートに対する各社の回答の密度にばらつきがあるように感じられることである。企業秘密がからむ技術問題だけに難しいが，ひとつの対策は，デルファイ法のように，集約結果を回答者に返して見直しを求めることであり，今回もこれを考えてみたい。

「日本的ソフトウェア工学の追求」に関しては札幌大学からの調査研究費支給が確定しており，今年の夏，シカゴ地区，サンフランシスコ地区において米国のソフトウェア生産現場を調査する予定である。

なお本研究は札幌大学産業経営研究所の1990年度予算を受けて行われた。また研究の一部には北海道科学研究費補助金の交付を受けた。

最後に本研究でお世話になった方々に，心からお礼申し上げます。

参　考　文　献

(1) 大野・松本　構造化ソフトウェア開発システム　ソフトウェア工学 40-2（1985．2．7）

表. 2 アンケート項目一覧

(1) アンケートには趣旨を説明し協力を依頼するかがみを付したがここでは省略した。
(2) アンケートの各項目には狙いを理解して頂くための若干のコメントを付しているが，原文にはこれはない。アンケート実施に当たっては，最初に依頼先を訪問し，記入要領についての説明を行っている。

Ⅰ．企業の諸特性はソフトウェア生産技術にさまざまな影響を及ぼすと考え，以下の質問を致します。可能な範囲でお答え下さい。

1．従業員数

(1)総計 ＿＿＿＿＿＿名
内 (2)営業要員 ＿＿＿＿＿＿名
(3)システム技術スタッフ（OS，開発環境などを担当） ＿＿＿＿＿＿名
(4)システムアナリスト ＿＿＿＿＿＿名
(5) SE ＿＿＿＿＿＿名
(6)プログラマ ＿＿＿＿＿＿名

その他補足事項がありましたらご記入下さい。

注．ここでは経営規模，経営体質が明らかになることを期待している。健全な企業では営業要員，技術スタッフ，その他（システム開発）要員がバランスしている筈で，企業の特質が各種要員の比重に表われる筈である。

従業員分類の定義は必ずしも明確ではない。役員，アズバイト要員，他社からの派遣要員などの数え方，SEやシステムアナリストの定義など，各社で異なり得る。

2．取引先

貴社の取引先に関し，下記該当項目に○印をご記入下さい。

( ) 一般マーケット
( ) 特定のコンピュータメーカーとの取り引きが大半
( ) 特定のコンピュータメーカーと一般マーケットとが相半ば
( ) 特定のユーザ企業との取り引きが大半
( ) 特定のユーザ企業と一般マーケットとが相半ば
( ) 一般マーケットが大半

その他補足事項がありましたらご記入下さい。

注．各社のマーケットを大まかに把握するのが狙いである。一般マーケット，特定のコンピュータメーカー，特定のユーザ企業について，売上げ高の比率を示してもらうべきだったかも知れない。

3．プロジェクトの規模

取引に関して行われる貴社のプロジェクトの平均規模はどの程度でしょうか。該当する項目に○印をご記入下さい。

3.1 プロジェクトチームのメンバー数

( ) 3人以下
( ) 4人ないし7人
( ) 8人ないし11人

（ ）12人ないし15人
（ ）16人ないし20人
（ ）21人以上
注．プロジェクト規模が役立つデータとなることを期待しての質問であるが，プロジェクトの工期もたずねるべきだったかも知れない。

3.2　開発するシステムのプログラムサイズ
（ ）　2000行以下
（ ）　2000ないし　5000行
（ ）　5000ないし 10000行
（ ） 10000ないし 50000行
（ ） 50000ないし100000行
（ ）100000行以上
その他補足事項がありましたらご記入下さい。

4．プログラム言語
貴社で用いておられるプログラム言語に○印をご記入下さい。
（ ）COBOL　　（ ）C
（ ）PL1　　（ ）APL
（ ）FORTRAN　　（ ）PASCAL
（ ）ASSENBLER　　（ ）BASIC
（ ）その他
その他補足事項がありましたらご記入下さい。
注．第4世代言語の利用状況が興味深い。

5．貴社で開発される主要なソフトウェア類型について，該当項目に○印をご記入下さい。

5.1　用途
（ ）業務用
（ ）科学技術計算
（ ）CAD/CAM用
（ ）システムソフト（OS, DATA BASE MANAGER, ETC）
（ ）通信用ソフトウェア
（ ）その他
注．本調査では主として業務用を対象としているが，その他の用途のソフトウェア開発が企業体質にどんな影響を及ぼすかは興味深いことである。

5.2　システムタイプ
（ ）バッチシステム
（ ）リアルタイムシステム
（ ）その他
その他補足事項がありましたらご記入下さい。

II．詳細に入る前に，経営上の技術的問題点をマクロ的にお尋ね致します。ソフトウェア開発プロジェクト

の各フェーズにおいて日頃遭遇している問題点と現時点で考えておられる解決策を簡単にご記入の上，問題の大きさの順位を（ ）内に記入して下さい。

1．販売促進活動 （順位＝ ）
（問題点）
（解決策）
注．販売力は技術力，製造力とならんで企業経営の要である。ここがどこまで強化され，問題意識がどこまで高まっているかは，企業体質を判断するうえで重要である。

2．リソースの見積り （順位＝ ）
（問題点）
（解決策）
注．ソフトウェア開発を受注するに際しての見積り精度は，受注者にとって致命的に重要であるが，今日の開発コスト，開発工期の見積り精度は極めてよくない。ソフトウェア企業がこの問題にどう対処しているかは興味深い。

3．要求仕様の確定 （順位＝ ）
（問題点）
（解決策）
注．ソフトウェア開発の問題点の多くは，「何を作るか」正しく決められないことに起因している。ソフトウェア工学の貢献が最も望まれるフェーズである。

4．システム設計 （順位＝ ）
（問題点）
（解決策）
注．ソフトウェアの効率的生産のポイントは既存ソフトウェアの再利用である。システム構造を歪めずに既存のプログラム構造やモジュール等をどう活かすかは，ソフトウェア企業の技術力を最もよく反映する筈である。

5．プログラム設計 （順位＝ ）
（問題点）
（解決策）
注．ソフトウェア再利用を的確に行うには，個々のモジュールの構造化が徹底していなければならない。この点についての問題意識は重要である。

6．コーディング・単体テスト （順位＝ ）
（問題点）
（解決策）

7．システムテスト （順位＝ ）
（問題点）
（解決策）
注．多くの企業で行われているボトムアップシステムテストは，大きな問題があとに残るという重大な欠点をかかえる。この点がどこまで認識されているか興味深い。

8．インストール・引渡テスト（順位＝　）
（問題点）
（解決策）
注．引渡テストの形態は商取り引きの完成度を表す指標となる。「構造化仕様書は引渡テストのテストケースである」という理想形態がどこまで確立されているか。

9．システムメンテナンス　（順位＝　）
（問題点）
（解決策）
注．ソフトウェアユーザが自分の手でソフトウェア機能の追加変更を行うのは，ソフトウェアの近代的生産財としての後進性を表す象徴的事象である。この点がどこまで認識されているだろうか。

10．プロジェクト管理　（順位＝　）
（問題点）
（解決策）
注．プロジェクトの成否はプロジェクトリーダーの手腕に決定的に依存する。すなわちプロジェクト管理の重要性は極めて大きい。この点に各企業はどう対応しているだろうか。

11．プロジェクト品質保証　（順位＝　）
（問題点）
（解決策）
注．プロジェクト品質保証においては，プロジェクト品質をどう定義するかが最初のステップである。今日の実態としては，ソフトウェアの取り引きでは品質保証まで手が回っていない。

12．プロジェクト納期保証　（順位＝　）
（問題点）
（解決策）
注．今日のソフトウェアの納期は，進歩の早い産業界の実情に全く合わない。この点に各企業がどう対処しているかは興味深い。

13．技術蓄積と再利用　（順位＝　）
（問題点）
（解決策）
注．ソフトウェア技術の蓄積はソフトウェアの再利用に直結する問題であり，各社が最大の関心を寄せる点である。

14．社員教育　（順位＝　）
（問題点）
（解決策）
注．「日本のソフトウェア業界は素人の集まりである」とも言える。大学の文学部を卒業した者が入社後立派に成長していく現状に問題はないのだろうか。大学はこの領域で何をなすべきなのだろうか。

15．新製品開発・技術開発　（順位＝　）
（問題点）

（解決策）
注．個々の企業がここにどこまで手をまわしているだろうか。

III．ソフトウェア工学的技術活用状況

ソフトウェア開発プロジェクトの各フェーズで活用されている技術とその具体的効果についてお聞かせ下さい。フェーズごとに列挙されている技術の中から貴社で活用されているものを選んで番号に○印を付し，それ以外の技術で活用されているものについてはその他の項目に記入し，それらの効果を効果欄に簡明に記述して下さい。

1．販売促進活動

1．その他（　　）　2．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。
注．販売促進活動の決め手となるような方法論は，現在のところ見当たらない。

2．リソースの見積り

1．リソースモデル　　2．ファンクションポイント
3．その他（　　）　4．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。
注．リソース見積りの問題点は業務要素を積み上げても精度が得られないということであり，一般のリソースモデルは有効ではないように思われる。ファンクションポイントはソフトウェア規模の表現方法であり，積み上げ不可という問題を越えることはできない。

3．要求仕様の確定

1．ユーザによる要求仕様の作成　　2．BSP
3．構造化分析　　4．エンティティリレーション分析
5．プロトタイピング　　6．デシジョンテーブル
7．HIPO　　8．SADT，SREM 等
9．ワークデザイン　　10．CASE ツール
11．その他（　　）　12．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。
注．構造化分析，エンティティリレーション分析，プロトタイピングなどは何を作るかが正しく決め難いという問題を解決する方法論として注目されるものである。ソフトウェアの健全な商取り引きを確立しようとするとき，この領域が最大の焦点になる。

4．システム設計

1．構造化設計　　2．ジャクソン法
3．ワーニエ法　　4．HIPO
5．情報の隠蔽　　6．システム構造のパターン化
7．部品化　　8．CASE ツール
9．設計査閲　　10．ウォークスルー
11．その他（　　）　　12．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。

注．アメリカでは（ジャクソン法，ワーニエ法が殆ど使われないのに比して）構造化設計が広く使われていると報告されるが，これは単なるモジュール化と同一視されている恐れがある。モジュールの再利用を真剣に考えるとき，情報の隠蔽は重要な技術となる。データベース設計作業もこのフェーズに入る。

5．プログラム設計

1．プログラム構造のパターン化　　2．PAD，HCP などの設計図法
3．フローチャート　　4．疑似コード
5．デシジョンテーブル　　6．自動支援ツール
7．その他（　　）　　8．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。

注．プログラム構造のパターン化はパターンの登録，コーディング変更箇所の指定等の管理の厳格さにおいて，かなりの幅があり得る。PAD，HCP 等のプログラム設計図法の採用はプログラムの構造化を必然ならしめるので，この普及状況には興味が持てる。これらの設計図を完璧に書けば，これを自動的にコードに変換することは容易であるが，プログラム設計者がこれをどこまで詳細に書くかは，コーダーとの関係において問題をはらむ。

6．コーディング・単体テスト

1．構造化プログラミング　　2．プログラム設計図の自動変換
3．第 4 世代言語（開発用）　　4．チーフプログラマチーム
5．その他の自動変換ツール　　6．その他（　　）
7．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。

注．第 4 世代言語はコボル等の第 3 世代言語にとって代わるという見解が一部の人たちから出されて久しく，その動向が注目されるところである。

7．システムテスト

1．構造化ウォークスルー　　2．コード査閲
3．トップダウンテスト　　4．第三者テスト
5．テストデータジェネレータ　　6．その他の自動支援ツール
7．その他（　　）　　8．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたら記入して下さい。
注．プロジェクト全体に占めるシステムテストの比率は極めて大きい。真剣に合理化に取り組んでいる企業なら，このフェーズに何らかの工夫をしていることが予想される。構造化ウォークスルーは，プログラムを個人の製品からチームの製品に変えるという点で重要であるが，これを真に効果的ならしめるには工夫が必要かもしれない。

8．インストール・引渡テスト

1．引渡テストケースの開発体制　　2．テスト結果の自動比較判定
3．その他（　　）　　4．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたらご記入下さい。

9．システムメンテナンス

1．ソフトウェア資産管理　　2．クロスレファレンス
3．構造化プログラムへの自動変換ツール　　4．その他（　　）
5．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたらご記入下さい。
注．ソフトウェア資産の管理を行っていないユーザはいないが，ここでの資産管理はモジュール単位の履歴を厳密に保存し，事故の多いモジュールは積極的に作り変えるというレベルの管理である。非構造化プログラムを自動的に構造化する“エンジン”の開発はいろいろ報告されているが，実際に使われているのだろうか。

10．プロジェクト管理

1．Steering Committee　　2．ウォーターフォール型ライフサイクル
3．その他のライフサイクル　　4．プロジェクト管理自動化支援ツール
5．PERT/CPM　　6．レビュー
7．その他（　　）　　8．その他（　　）
効果（NO.　）
効果（NO.　）
その他補足事項がありましたらご記入下さい。
注．プロジェクトライフサイクルは個々のプロジェクトが自分に最も適切なライフサイクルを設計し，それにそって業務を展開する業務計画書である。ライフサイクルの設計には過去の蓄積が極めて重要であるが，

これがどこまで行われているだろうか。

Ⅳ．その他

以上の調査に対して補足すべき事項がありましたら記述して下さい。

表．3　アンケート結果

(1) 記入された事項についての若干の注記を付した。

Ⅰ．企業の諸特性はソフトウェア生産技術にさまざまな影響を及ぼすと考え，以下の質問を致します。可能な範囲でお答え下さい。

1．従業員数

(1) 総計

A：（2000）	D： 83
B：544	E：354
C：320	F： 43

内 (2) 営業要員

A：（200）	D： 6
B：16	E：22
C： 3	F： 2

注．A社の営業要因のほとんどはSE出身であるが，科学技術分野の営業には営業プロパーとして育った営業要員が数名いる。数字が括弧でくくってあるのは，この数字が直接アンケートに記入されたものでなく，面談の結果を記入したものであることを示す。

B社は大手企業の支社であり，本社にかなりの営業要員が配置されている。

(3) システム技術スタッフ（OS，開発環境などを担当）

A：	D： 1
B：44	E：14
C： 3	F： 1

(4) システムアナリスト

A：	D：11
B：15	E：29
C：（SEとプールしている）	F： 1

(5) SE

A：	D： 35
B： 83	E：117
C： 44	F： 19

(6) プログラマ

A：	D： 27
B：156	E：172
C：176	F： 16

注．役員，アルバイトなどの扱い方に不統一があり得る。また技術スタッフ，システムアナリスト，SE，プログラマなどの区分は必ずしも明確でない。

2．取引先

A：（一般マーケット）
B：特定のユーザ企業と一般マーケットとが相半ば
C：特定のユーザ企業および特定のコンピュータメーカー
D：特定のコンピュータメーカーおよび特定のユーザ企業
E：特定のユーザ企業と一般マーケット，特定のコンピュータメーカー
F：特定のソフトウェアハウスとの取引が大半
注．企業の中には大手ユーザから分社したものもあるが，「特定のユーザ企業」は必ずしも分社した親企業を示すものではない。

3．プロジェクトの規模

3.1 プロジェクトチームのメンバー数

A：
B：特定できない
C：4人ないし7人
D：4人ないし7人
E：特定のユーザ企業に対しては4人ないし7人（メンテナンスの場合），あるいは21人以上（新規プロジェクト），一般マーケットおよび特定のコンピュータメーカーの場合は16人ないし20人
F：8人ないし11人（プロジェクト全体では21人以上）
注．開発プロジェクトの中には複数社からの要員の派遣を得てチームを編成するものがあり，上記数字がそのままプロジェクトの規模を表わすとは限らない。

3.2 開発するシステムのプログラムサイズ

A：
B：特定できない
C：10000ないし50000行
D：10000ないし50000行
E：特定のユーザ企業に対しては4人ないし7人10000ないし50000行（メンテナンス）100000行以上（新規）一般マーケット・特定のコンピュータメーカーの場合は100000行以上
F：10000ないし50000行（プロジェクト全体では100000行以上）

4．プログラム言語

A：（各種）
B：COBOL, PL1, ASSEMBLER, C, BASIC
C：COBOL, ASSEMBLER, C, BASIC
D：COBOL, PL1, ASSEMBLER, C, BASIC
E：COBOL, PL1, FORTRAN, ASSEMBLER, C
F：COBOL, PL1, ASSEMBLER, C, 他にCSP, RPGなどの簡易言語

5．貴社で開発される主要なソフトウェア類型について，該当項目に○印をご記入下さい。

5.1　用途

A：（各種）
B：業務用
C：業務用
D：業務用，CAD/CM 用，システムソフト（OS，DATA BASE MANAGER，ETC）
E：業務用
F：業務用，CAD/CAM 用，ソフトウェア開発支援ツール

5.2　システムタイプ

A：（各種）
B：バッチシステム，リアルタイムシステム
C：リアルタイムシステム
D：バッチシステム，リアルタイムシステム
E：バッチシステム，リアルタイムシステム，TSS
F：バッチシステム，リアルタイムシステム

II．詳細に入る前に，経営上の技術的問題点をマクロ的にお尋ね致します。ソフトウェア開発プロジェクトの各フェーズにおいて日頃遭遇している問題点と現時点で考えておられる解決策を簡単にご記入の上，問題の大きさの順位を（　）内に記入して下さい。

1．販売促進活動

A：
B：マーケッティング，開発，技術を有機的に結合した計画主導の業務推進体制の確立が急がれる中で，マーケッティングの概念が不確かで，マーケッティングを担う人材の層が薄い（順位1）。
　業務診断プロジェクトの積み重ねをマーケッティングの基盤にすることを基本に，人材の育成を進めている。
C：特に問題を感じていない。
D：大手メーカーとの取引が大半であるため特に問題はない。
E：複数社が参加するプロジェクトの受注において，他社とのインターフェースの多さを加味した難易度を見極めに苦慮している。
　解決策としては，現在は経験則の蓄積しかない。
F：販売活動は特に行っていない。
注．各項目の重要さの順位づけも含めた記述の詳細さは，企業間でかなり異なる。この理由として（1）企業としての問題意識の深さ（2）アンケートにどこまで詳細に応じるかについての方針等に由来するものと思われる。これに対処するための一法として，デルファイ法のようにアンケートの最初の集約結果を参考にしての再度の記入を依頼することが考えられる。

2．リソースの見積り

A：

B：正確な見積りを可能にする要求定義技術がなく，本来は一部将来物理モデルまで作成して可能となる見積りを基本構想段階で行わねばならない（順位6）。
過去の実績に基づく見積りが要素の積み上げよりも精度が高いのが現状であるが，新見積り技術の開発，見積り精度の現状に見合った契約方法なども検討している。

C：契約の基本になる工数見積りに充分な工数をかけることができない。
見積り基準の改定を行うべきであるが，ユーザとの関係において難航が予想される。

D：稼働環境に応じての多様な開発環境が必要になり，不都合を感じる。
今のところ何とか耐えられる状態ではある。

E：大型コンピュータのみでなく，オフィスコンピュータの見積りも大変である。見積りを複雑にする要因として，業務自体の複雑さ，開発環境の多様性，簡易言語，開発技術（例えばパターン化・部品化技術の評価）など，システム規模と所要工数の関係を左右する多くの項目が挙げられる。
解決策として，機能および開発環境を重視した所要工数算定方法を検討している。

F：大規模プロジェクトにおいては，プロジェクトの発注もとであるコンピュータメーカーの営業担当者あるいはSEから見積りが提案される場合が多いが，要求定義の精度によって見積りに大きな狂いが生じる（順位5）。
対策として，基本設計あるいは詳細設計以降で契約する場合，請負工数精算方式によってリスク回避を行うこともある。

3．要求仕様の確定

A：

B：客先の責任体制や意思決定支援ルートがあいまいで時間がかかり，また明確な要求仕様が定まらない，要求仕様の変更追加が少なくないなどの問題がある。
解決策としては，客先の要求仕様決定プロセスを明確にしてもらう，ウォークスルーを強化する，要求仕様変更時の契約上の処理の明確化，上流SEの業務知識の拡大，保有する要求仕様モデルの部分的変更による要求仕様の確定などを検討推進している。

C：ユーザの要件が開発に入ってから変更されることが多い。
開発に入る前の打合せの充実に心掛けている。

D：自社独自の仕様の確定・承認体制がない。現状は取引先ごとに基準を決めている。
自社内の承認体制だけは明確にしたいと考えている。

E：・特定のユーザを相手としているが，ユーザ側の力量が次第に失われ，要求仕様確定における負担が増加してきている。
SEの業務知識の強化，要求仕様確定手順書の充実，構造化分析ツールの活用など検討している。
・要求仕様が必要充分であるか否かに確信が持てない，システム知識の充分でないユーザにも受け入れられる表記法を欠く，過去の類似要求仕様の再利用が行われていないなどの問題もある。
要求仕様の文書化，標準化，パターン・部品化を進めること，DFDなどユーザに理解し易い表記法の採用など検討している。

F：ユーザがコンピュータシステムに対する基本的知識に欠け，完成時のシステムイメージを概観できぬままに曖昧な要求仕様を提示する，ユーザ業務についてのSEの知識が不十分である，業務を発注するメーカーの営業担当者，SEが旨い話をしたがるなどの要因が重なって，曖昧な要求仕様，楽観的な工程表が作られる（順位3）。
要求仕様の不備によるシステムの修正にはコストがかかることをユーザに理解させ，また自社の営業対象業務を絞って，業務によく通じたSEを育てるなどを考えている。

4．システム設計

A：
B：システム構造の設計のための実用的で分かりやすい方法論がなく，経験やカンに頼っている（順位5）。
理解と経験を融合させ得る方法論を模索している。
C：特になし。
D：顧客ごとに設計標準が定められているが，これに対する自社の検査基準，承認体制が明確でない。
現在はプロジェクト内のチェックと発注先の承認により次工程に進んでいる。
E：設計技術者の不足が基本的問題であり，設計書の品質評価が行いにくい，過去の設計例の再利用が行われていない，分散化等の新しい技術動向に対する技術蓄積が不十分であるなどの問題も重大である。
設計手順書の整備と技術者教育の推進，ウォークスルーの強化，過去の経験のカプセル化（文書化，パターン化，部品化など）等を検討推進している。
F：（1）設計思想の統一が困難（2）複数のソフトウェア企業で構成するプロジェクトにおいてはソフトウェア企業ごとの人員，能力に対応しての不自然なサブシステム分割がなされサブシステム間の不整合を招きやすい（3）ドキュメントの作成・修正に要する負荷が膨大であるなどの問題がある（順位4）。
（1）に対しては設計標準書およびレビューの強化（2）に対してはより適切なサブシステム分割とサブシステム間インターフェース会議の強化（3）については自動化ツールの導入を進めたい。

5．プログラム設計

A：
B：プログラム仕様書の書き方の標準化さえ不徹底である。外注先の新人教育のための負荷が大きい（順位12）。
標準書の強化と徹底（プログラマがSEに内容の説明を行う）を進めている。
C：特になし
D：システム設計と同様
E：設計書の評価が行いにくい，プログラム仕様書がそのままドキュメントとしては使えない，設計仕様書をどの程度の詳細さで書くべきか（プログラマの果たすべき役割）などの問題を抱える。
プログラムのパターン化・部品化，最終ドキュメントを意識したツールの導入，機能表記法の改善，ウォークスルーの充実などを対策として考えている。
F：設計者により内容，精度などに差がでる，ドキュメントの作成・修正に時間がかかるなどの問題がある（順位8）。
標準化の強化，スケルトンロジックの準備，設計書レビューの強化，自動化ツールの導入などを考えている。

6．コーディング・単体テスト

A：
B：人による能力差が大きい（順位13）。
図式仕様記述からコードを自動生成することを指向し，また制作技術グループを発足させて技術レベルの向上をはかることを実行している。
C：コードの標準化が進まず，また単体テストが充分に行われていない（見積り工数も小さい）。
教育の強化，見積り基準の改善を考えている。
D：システム設計と同様

E：コードの標準化が充分でない，単体テストの品質が不十分（複合条件テスト，論理的誤りの見逃し）などの問題がある。
コードのパターン化，コードの自動生成，デバッガー機能の強化，チーフプログラマチームの採用などの対策を検討している。
F：設計者とコーダーの意志疎通，テストケース不十分，難解なプログラムの制作などの問題がある（順位9）。
設計者とコーダーのミーティング，テストケース表の作成とウォークスルー，スケルトンプログラムの準備とコーディング規約の強化などを考えている。

7．システムテスト

A：
B：非常に長時間を要し，テストデータ準備の負荷が大きい（順位14）。
プロジェクトの早い段階から並行的にテストの行える環境を準備し，またデータ生成系のプログラムのテストを先行させてテストデータ生成の手間を省くことを考えている。
C：実稼働と同じ環境が準備できず，テストの質が充分でない（順位7）。
対策としてはテストマシンを強化するしかない。
D：システム設計と同様
E：「要求仕様書＝テストケース」という体系が確立されておらず，エンドユーザによる直接テストに依存するケースが生じる。
要求仕様書を定型化し，これに基づくテストが開発サイドのみで行える状態を目指したい。
F：前工程の遅延で質の高いテストが行えない，この段階でユーザからの仕様変更が多発する，立会待機でプログラマの士気が低下するなどの問題がある（順位6）。
楽観的なスケジュールを排し必要に応じてスケジュール変更を行う，プロトタイピングの実施，テスト方法の改善等が必要である。

8．インストール・引渡テスト

A：
B：システム停止即生産停止につながる大規模システムにおいてインストール時間を極力短縮することの難しさ，運用習熟環境設定の難しさ（順位15）。
詳細な計画書を作成し完璧なインストール作業を行うことが対策となる。
C：特になし。
D：仕様書設計と同様
E：特になし。
F：休日，夜間などでのインストールが要求され，移行漏れの問題もおきる（順位10）。
前者はやむを得ず，後者についてはプログラムの自動管理を考えている。

9．システムメンテナンス

A：
B：ドキュメント管理不十分，メンテナンスの影響範囲の特定に手間取る，要員がメンテナンス作業に固定されローテーションが困難である，などの問題がある（順位11）。
ドキュメント管理の徹底，システム構造の改善，計画的な要員配置と業務研修などの対策が考えられる。

C：ドキュメントのない古いプログラムの存在（順位５）。
具体的な解決策に窮している。
D：エンドユーザの窓口抜きの直接のやりとりとなっていて，けじめがつかない。
ソフトウェア引渡後はメンテナンス業務の窓口が必要である。
E：ドキュメントの不備不統一(制作用と保守用では必要なドキュメントが異なる)，影響範囲の特定化困難，アセンブラプログラムの混在，メンテナンスによるシステム構造の乱れ，要員ローテーションの難しさ（KNOW-HOWの非文書化）など。
メンテナンス支援ツールの導入(含ドキュメント支援機能)，古いプログラムのリプレース，教育などが必要である。
F：ドキュメントのメンテナンスが行われていない，拡張性に欠けるシステム，メンテナンス要員の士気低下など。
ドキュメントの自動管理化，システム設計の改善，要員のローテーションなどの対策が必要である。

10. プロジェクト管理

A：
B：プロジェクトリーダー，プロジェクトメンバーともに，プロジェクト管理に関する共通認識に乏しく，実行力も不足している（順位２）。
新人教育の段階からプロジェクト管理の基礎を徹底，プロジェクト毎にプロジェクト技術管理者を任命しプロジェクト技術管理の定期的討論会を開催，プロジェクト推進会議の定期的開催などを実行している。
C：プロジェクトを充分に管理できる人材が不足している（順位６）。
教育とローテーションとでプロジェクト管理者の育成をはかっている。
D：プロジェクトごとに管理方法が異なり，また自社基準がないため，プロジェクトにおける自社の責任が不明瞭になりがちである。
対策として他の複数企業と参画するプロジェクトも包含した自社のプロジェクト管理基準の作成が急務である。
E：大規模プロジェクトの経験者が少なく，プロジェクト管理者と若手の間の経験的ギャップが大きい。
プロジェクト管理支援ツールを導入し大規模プロジェクト実績の蓄積を行っている，また大規模プロジェクトを教材とした教育を計画している。
F：ビジネス系の大規模プロジェクトでは元請企業がプロジェクト管理を行っているが，要件確定が甘く，仕様変更要求を容易に受入れ，進捗把握が甘く，工程変更が多発しがちである（順位11)。
要件確定作業にパターン化，プロトタイピングなどの技術を持ち込んで技術水準向上をはかり，また進捗把握方法の改善を行うなどの工夫が必要である。

11. プロジェクト品質保証

A：
B：機能保証とバグの管理が精一杯で，システム構造，操作性，解読性，性能などを包含したシステム品質全体の保証活動までは行われていない（順位７）。
システム構造の方法論確立から着手し，逐次体制を整える。
C：品質管理のための統一的な制度が設定されておらず，品質管理は各プロジェクトに任された状態で，充分な品質が保証されているとは言えない（順位１）。
保証体制の制度化と要員の配置が必要である。

D：特になし。
E：上流工程での品質確保，プログラム品質の検査方法の確立が課題である。
ウォークスルー，フェーズド・アプローチの徹底，バグ抽出件数を目安とした管理体制の確立を進めている。
F：特になし。

12. プロジェクト納期保証

A：
B：見積り不備，環境変化，プロジェクト管理不徹底などで納期遅延を招くことがある。
また納期達成時にも高負荷となることが多い（順位8）。
原因となる部分の改善を行うことが対策となる。
C：特になし。
D：適正な納期を設定するために，ユーザとの打合せに開発要員が直接でむいている状態である。
営業体制の充実強化が急がれねばならない。
E：品質の評価も含めた進捗管理が行われなければ，納期の的確な保証はできない。
フェーズド・アプローチ，ウォークスルーなどが対策となるが，上流工程に関しては決め手にならない。
F：特になし。

13. 技術蓄積と再利用

A：
B：システム機能，システム構造，開発プロセスを技術蓄積のためにモデル化しているプロジェクトはほとんどない（順位9）。
システム構造設計の方法論，プロジェクト管理技術，形式的仕様記述等の基盤技術を徹底し，この上に組織的な技術蓄積活動を展開すべく，一部着手している。
C：特になし。
D：技術の蓄積は個人情報として留まっており，それを組織的に収集蓄積して活用する機会がほとんど持てない。
解決策としては，技術情報を収集管理するための専門組織の設置が考えられる。対象業種，マシンを特化すれば，効果的な蓄積が可能である。
E：技術蓄積が個人の頭の中で行われている比重が大きく，設計書からコードまでの再利用が充分に行われていない。
個人技術の文書化と文書管理支援ツールの導入，文書，コードのパターン化，部品化などの推進を検討している。
F：技術が個人的に蓄積され，企業としては残らず，また開発環境，技術標準の差異，著作権，ドキュメント化の不備等で再利用がほとんどなされていない（順位2）。
知識のデータベース化，（コードでなく）ロジックの再利用を検討したい。

14. 社員教育

A：
B：学校教育において情報化に関する基礎概念の形成が行われておらず，専門SE養成の阻害要因となっている。また新人教育でかなりのことをやっているが，配属後のOJTが著しく不備である（順位3）。

学校教育の改革が必要であり，システム分析，システム設計の方法論の徹底が基盤となる。各種講習会，シンポジウム，学会への参加，設計技術講座の開催などを行っている。

C：充分な教育体制がなく，その確立が急がれねばならない（順位８）。

D：経験が先行する実践教育に偏っている。
バランスのとれた企業人の育成が必要であり，情報処理技術者試験の活用，新人教育に始まる教育カリキュラムの見直しなどに着手している。

E：プロジェクトリーダー，上級SE(提案型SE)，SE，プログラマ等に分けた体系的教育が必要であるが，顧客の技術に合わせての技術教育も行わざるを得ない。

F：実務以外は個人の自主性に任せているが，実態はほとんど勉強していない（順位１）。
SE教育研修，国内外の技術研修への派遣，資格取得者への資格手当の支給など，自己啓発へのインセンティブの導入を図りたい。

15. 新製品開発・技術開発

A：

B：製品開発，技術開発を支えるべきマーケッティング力が不十分であり，これを支える技術基盤，人材の蓄積も乏しい（順位10)。
マーケッティング能力の向上，技術基盤，人材の育成を着実に進めるかたわら，先進企業との技術提携，異業種との交流等を推進することが必要である。

C：特になし。

D：特になし。

E：アセンブラから第３世代言語への自動変換技術，分散化のための基礎技術の必要性を痛感している。
技術調査と既存技術の導入を検討している。

F：特になし。

III. ソフトウェア工学的技術活用状況

ソフトウェア開発プロジェクトの各フェーズで活用されている技術とその具体的効果についてお聞かせ下さい。フェーズごとに列挙されている技術の中から貴社で活用されているものを選んで番号に○印を付し，それ以外の技術で活用されているものについてはその他の項目に記入し，それらの効果を効果欄に簡明に記述して下さい。

１．販売促進活動

A：

B：構造化分析技法，IE･QC技法を有効に活用している。これにより顧客の業務分析を的確に行い，受注に結び付けている。
構造化されたモデルパッケージを開発保持することが今後の重要な販売促進手段となると考え，今後の研究課題としている。

C：特になし。

D：特になし。

E：特になし。

F：特になし。

2．リソースの見積り

A：
B：過去の実績をデータベース化し，データフロー図とファンクションポイント法を試みなど行っているが決め手を欠いている。
C：特になし。
D：特になし。
E：ステップ数中心の見積りを行っているが使用プログラム言語，処理パターン等が変わると精度が落ちる。ファンクションポイント法は見積りがステップ数から機能数に変わるという点では期待は持てるがバックデータの蓄積が必要である。
F：ビジネス系では経験に基づく見積りや請負工数精算方式を採用しているが，エンジニアリング系ではフェーズごとの見積りをして見積り精度を高める場合もある。

3．要求仕様の確定

A：
B：構造化分析技法を要求仕様定義の基本技術として用いている。顧客に書いてもらうケースも多く，要求仕様に関する共通認識を得るのに効果的である。
エンティティリレーション分析も試みている。コンパクトな資料で全体像を概観するのに好都合であるが，作成負荷は大きい。
プロトタイピングも活用し期待通りの成果を得ている。
デシジョンテーブルを顧客に条件書として記入してもらい，要求仕様の明確化に役立てている。
CASE ツールは効率のよいワープロとして活用されている。
C：特になし。
D：特になし。
E：構造化分析を試行している。データフロー図はユーザと共通認識を持つためには役立つが，ユーザがシステムを理解するための充分な要素を持っているとは言えない。
HIPO は良く利用しているが，ユーザに理解してもらう資料にはなりにくい。
ユーザと一体になってのシステムシステム分析作業は有効であるが，顧客が遠隔地にある場合などには適用しがたい。
F：ユーザによる要求仕様作成を行っている。このような古典的方法を取らざるを得ない理由は，ユーザのコンピュータに関する知識，開発者の業務に関する知識が不足しているためである。

4．システム設計

A：
B：設計査閲，ウォークスルーは業務標準として規定され，システム品質の向上，システムの全体像やプロジェクトの進め方についてのメンバーの理解認識の共通化，メンバーの能力開発に大きな効果を挙げている。
HIPO はプロジェクトによっては適用され，設計仕様の明確化に効果を挙げている。
データフロー図を用いたシステム構造のパターン化，部品化は研究段階である。
C：オンラインの典型的な処理要素をパターン化して効果をあげている。
パターンとは別に標準部品の利用も行っているが，この多くはマクロとして準備されている。
D：特になし。

E：構造化設計，ウォークスルー，パターン化，部品化，HIPOなどを活用し，CASEツールを利用している。
構造化設計は適正な標準化を基盤に経験を活かして活用しているが，構造化設計技法として深く意識されてはいない。分かりやすいプログラムを作成する上では効果を挙げている。
ウォークスルーは上流工程の品質確保に有効であり，また遠隔の顧客とのレビューの効率を上げるのにも役立つ。
パターン化，部品化の活用は組織として正式に行っているものではなく，効果を高めるには文書の電子化が必要である。
HIPOは体系を一貫して利用しているわけではない。
CASEは画面入出力設計に有効である。
F：システム構造のパターン化を行うことにより，設計のバラツキを少なくし，設計作業量を削減することが可能になっている。

5．プログラム設計

A：
B：PAD，フローチャートを共に用いているが，前者の比重が高まり，プログラムの構造化に効果を挙げている。
プログラム構造のパターンは，プロジェクトによっては積極的に採用し，品質，生産性の面で大きな効果を得ている。
デンジョンテーブルは複雑な論理の表現に用いられ，効果を発揮している。
“DIAGRA”というCASEツールの適用が次第に拡大しており，PAD化，パターン化，部品化の促進に役立っている。
C：プログラム構造のパターン化はプログラム構造の標準化，プログラム品質の向上等に役立っている。
疑似コードも利用しているが，これはアセンブラのプリプロセサ用に変形し易いという利点がある。
D：特になし。
E：プログラム構造のパターン化，PAD，フローチャート等を用いている。
プログラム構造のパターン化は生産性，品質の向上，プログラム構造の統一に効果を発揮している。
PADはソースコードの自動生成ツールと合わせて用いられている。あいまいな設計の排除，プログラムの構造化，生産性向上に有効である。
F：プログラム構造のパターン化で設計者によるバラツキの減少，作業量の軽減などの効果を挙げている。
また設計図法の採用でロジックが理解し易くなり，エラーの検出が容易になった。

6．コーディング・単体テスト

A：
B：構造化プログラミングはPADの普及で徹底しつつある。
プログラム設計図の自動変換ツールの採用でコーディング，単体テストの生産性が大幅に向上している。
第4世代言語の採用は部分的であり，十分な効果を発揮するに至っていない。
C：構造化プログラミングの採用でコードの読みやすさが向上している。
D：特になし。
E：構造化プログラミングの徹底によりプログラムの可視性の向上に役立っている。
PADとソースプログラムの双方向自動変換可能なツールを用いており，生産性の向上やプログラムの理解しやすさの向上が実現されている。

また新人は職種転換者の戦力化のためにチーフプログラマチーム的なチーム編成を行うこともある。
F：構造化プログラミングによりメンテナンス性，拡張性の向上を実現している。
設計図とソースプログラム間の自動変換ツールも採用しており，プログラムと設計書間の食い違いの検出，プログラムと設計書の同期化が容易になっている。
また日本語COBOLの利用でプログラムのドキュメント性が向上している。

7．システムテスト

A：
B：テスト計画に対する構造化ウォークスルーの実施で，テスト精度の向上が得られている。
またユーザが自ら作成した運用マニュアルをもとにシステムの実端末テストを行うことにより，マニュアル，システム機能，プログラムの正さ，操作性等を一度に検証している（第3者テスト）。
C：特になし。
D：特になし。
E：第3者テストを実施しているが，負荷の大きさを勘案するとき，果たして本来の目的を達成しているか否か疑問である。
F：第3者により，先入観のない適切なテストケースを得ている。

8．インストール・引渡テスト

A：
B：ユーザとテストケースの相互確認を実施し，ユーザの参画意識の向上と機能の最終的な確認に効果を挙げている。
C：特になし。
D：特になし。
E：ファイルコンバージョンにおいては，必要に応じて結果の自動判定を行っている。
F：特になし。

9．システムメンテナンス

A：
B：ソフトウェア資産管理としてドキュメント管理体系を整備している。
またライブラリ内で利用しているタグ名称を探し出すサブルーチンを開発し，プログラムリストにクロスレファレンスを出力している。
C：特になし。
D：特になし。
E：プログラム台帳，プログラム関連図，プログラム仕様の厳格な管理を行っている。
またネーミングの標準化によってクロスレファレンスを可能にしている。
F：クロスレファレンスにより，調査，検索が容易になっている。

10．プロジェクト管理

A：
B："STEERING COMMITTEE"的なプロジェクト推進会議を実施し，各プロジェクトの管理レベルの確

認と向上に役立てている。
プロジェクトライフサイクルとしては過去10年ウォーターフォール型ライフサイクルを利用してきたが，最近プロトタイピングライフサイクルも用いている。
レビューはプロジェクト管理の基本的活動として位置づけている。

C：特になし。

D：特になし。

E：特になし。

F：進捗管理資料自動作成ツールを導入し，厳密な進捗管理を可能にしている。

IV. その他

以上の調査に対し補足すべき事項がありましたら記述して下さい。

A：

B：システム部門に「技術」，「工学」の概念がほとんど存在していないのがソフトウェア工学の成果が活用されない最大の理由である。技術，工学の概念欠如の最大の原因は学校教育においてこれがなされていないことにある。企業内教育ではソフトウェア工学の所産を整理学習するプロセスと，その成果をモディファイし組み合わせて応用するプロセスとを，確実に実行していくことが大切である。

C：特になし。

D：特になし。

E：ソフトウェア工学の成果がシステムの設計制作において十分に活用されない理由を次のように考える。
・理解が難しい
・設計，制作は基本的には人の真似をすることに依存している。実際面でも質のよい設計書やプログラムを真似ることで，かなりの精度が保証されている。
・特にパターン化，部品化が進めば，パターンや部品を開発する技術者以外は高度な技法を知らなくともよくなる。

F：新しい開発技術が生産現場で用いられない理由を次のように考える。
・ユーザ，開発技術者の教育不足
・過去の膨大な遺産
・管理者の保守性（リスク回避）
・コストがかかる
・複数のソフトウェア会社から技術者を寄せ集めるプロジェクト体制